09/872,054

Requested Patent: JP10247155A

Title: FILE SYSTEM INTERFACE TO A DATABASE ;

Abstracted Patent: EP0856803, A3 ;

Publication Date: 1998-08-05 ;

Inventor(s):

BALABINE IGOR V (US); SKIER JOHN A (US); KANDASAMY RAMIAH (US) ;

Applicant(s): INFORMIX SOFTWARE INC (US) ;

Application Number: EP19980300577 19980127 ;

Priority Number(s): US19970792139 19970131 ;

IPC Classification: G06F17/30 ;

Equivalents: AU5275798, AU739236, BR9800065, CA2228210, US5937406

ABSTRACT:

Information in a database is accessed with a computer system by transforming a file system request from an application into a database query and retrieving information corresponding to the database query from the database. The retrieved information is made available to the application as a file system object, for example, as a directory, a file, a link or a collection thereof.

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－247155

(43)公開日　平成10年(1998)９月14日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　12/00	５１２	Ｇ０６Ｆ　12/00	５１２
	５０５		５０５

審査請求　未請求　請求項の数50　ＯＬ　（全 16 頁）

(21)出願番号　特願平10－18637

(22)出願日　平成10年(1998)１月30日

(31)優先権主張番号　０８／７９２１３９

(32)優先日　1997年１月31日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　596158536
インフォアミックス　ソフトウェア　インコーポレイテッド
アメリカ合衆国　カリフォルニア州　94025　メンロ　パーク　ボハーノン　ドライヴ　4100

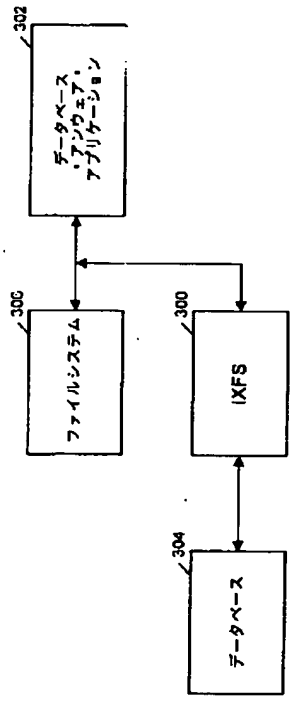
(72)発明者　イーゴー　ヴィー　バラピン
アメリカ合衆国　カリフォルニア州　95014　クーパーティノ　ベル　エア　コート　11063

(74)代理人　弁理士　中村　稔　（外６名）

最終頁に続く

(54)【発明の名称】　データベースへのファイルシステムインタフェース

(57)【要約】
【課題】　データベース内の情報にアクセスする方法及びシステムを提供する。
【解決手段】　データベース内の情報は、アプリケーションからのファイルシステム要求をデータベース照会に変形し、このデータベース照会に対応する情報をデータベースから検索することによって、コンピュータシステムによりアクセスされる。検索された情報は、アプリケーションが使用できるようにするために、例えばディレクトリ、ファイル、リンク、またはそれらの集まりのようなファイルシステムオブジェクトにされる。



【特許請求の範囲】

【請求項１】　データベース内の情報にアクセスするためにコンピュータシステム上で遂行される方法であって、

データベースオブジェクトを、アプリケーションがファイルシステムオブジェクトとして使用可能にするステップと、

上記使用可能なファイルシステムオブジェクトに対応するファイルシステム要求を、上記アプリケーションから受信するステップと、

上記アプリケーションからの上記ファイルシステム要求を、データベース動作に変形するステップと、を備えていることを特徴とする方法。

【請求項２】　上記データベース動作を、データベースエンジンを使用して上記データベース上で遂行するステップを更に備えている請求項１に記載の方法。

【請求項３】　上記データベース動作に応答し、上記データベースから情報を検索するステップを更に備えている請求項１に記載の方法。

【請求項４】　上記データベースオブジェクトに関連する情報を、ファイルシステムオブジェクトとして上記アプリケーションに戻すステップを更に備えている請求項１に記載の方法。

【請求項５】　上記戻しステップは、検索された情報を、拡張モジュール内に定義されているフォーマットに配列することからなる請求項４に記載の方法。

【請求項６】　上記フォーマットされた情報を、コンピュータの表示画面上に表示するステップを更に備えている請求項５に記載の方法。

【請求項７】　上記データベースオブジェクトに関連する情報を、コンピュータの表示画面上に表示するステップを更に備えている請求項１に記載の方法。

【請求項８】　上記データベースオブジェクトに関連する情報は、ファイルシステムオブジェクトの図形表現として表示される請求項７に記載の方法。

【請求項９】　上記ファイルシステム要求を発行する上記アプリケーションは、データベース・アンアウェアアプリケーションからなる請求項１に記載の方法。

【請求項１０】　上記データベースは、関係型データベースからなる請求項１に記載の方法。

【請求項１１】　上記データベースは、オブジェクト・関係型データベースからなる請求項１に記載の方法。

【請求項１２】　上記データベースは、オブジェクト指向データベースからなる請求項１に記載の方法。

【請求項１３】　上記変形ステップは、上記ファイルシステム要求を、データベース照会に変換することからなる請求項１に記載の方法。

【請求項１４】　上記データベース照会は、ＳＱＬコンプライアント照会からなる請求項１３に記載の方法。

【請求項１５】　上記変形ステップは、拡張モジュール内に収納されている情報に基づいて、上記ファイルシステム要求をデータベース照会に変換することからなる請求項１に記載の方法。

【請求項１６】　上記受信したファイルシステム要求が、ファイルシステムによって管理されている情報と一致するか否かを決定するステップを更に備えている請求項１に記載の方法。

【請求項１７】　もし上記ファイルシステム要求が上記ファイルシステムによって管理されている情報と一致することが決定されれば、上記ファイルシステム要求を上記ファイルシステムへ輸送するステップを更に備えている請求項１６に記載の方法。

【請求項１８】　上記決定ステップは、上記ファイルシステム要求が導かれるネーム空間を識別することからなる請求項１６に記載の方法。

【請求項１９】　上記決定ステップは、上記コンピュータシステムに関連するオペレーティングシステムの核アドレス空間において実行される方法によって遂行される請求項１６に記載の方法。

【請求項２０】　上記決定ステップは、上記コンピュータシステムに関連するオペレーティングシステムの外部において実行される方法によって遂行される請求項１６に記載の方法。

【請求項２１】　上記データベースオブジェクトをファイルシステムオブジェクトとして使用可能にするステップは、上記アプリケーションに対して透過的に遂行される請求項１に記載の方法。

【請求項２２】　上記データベースオブジェクトは、表からなる請求項１に記載の方法。

【請求項２３】　上記ファイルシステムオブジェクトは、ディレクトリ、ファイル、またはリンクからなる請求項１に記載の方法。

【請求項２４】　上記データベースオブジェクトをファイルシステムオブジェクトとして使用可能にするステップは、上記データベースオブジェクトを、複数のファイルシステムオブジェクトとして異なるアプリケーションによって理解可能なフォーマットで提示することからなる請求項１に記載の方法。

【請求項２５】　データベース内の情報にアクセスするコンピュータをベースとする方法であって、

上記データベースから情報を検索するステップと、

上記検索された情報を、ファイルシステムオブジェクトとしてアプリケーションに提示するステップと、を備えていることを特徴とする方法。

【請求項２６】　上記データベース内の複数のデータベースオブジェクトの中から１つのデータベースオブジェクトを指定することによって、上記データベースから検索される情報を識別するステップを更に備えている請求項２５に記載の方法。

【請求項２７】　上記識別ステップは、データベースオ

ブジェクトの所定の集合を記述する情報を、拡張モジュール内に記録することからなる請求項２６に記載の方法。

【請求項２８】　上記検索ステップは、上記拡張モジュール内に収納されている情報に基づいて、上記データベースから情報を選択的に抽出することからなる請求項２７に記載の方法。

【請求項２９】　上記提示ステップは、上記拡張モジュール内に収納されている情報に基づいて、上記選択的に抽出された情報をファイルシステムオブジェクトにフォーマットすることからなる請求項２７に記載の方法。

【請求項３０】　上記ファイルシステムオブジェクトの集まりが、ディレクトリ、ファイル、リンク、またはそれらの組合わせを有する階層的ファイルシステムを表している請求項２５に記載の方法。

【請求項３１】　コンピュータをベースとするデータリポジトリ管理システムであって、
情報のデータベースと、
データを操作するためのファイルシステムをベースとするアプリケーションプログラムと、
上記ファイルシステムをベースとするアプリケーションに上記データベース内の情報へのアクセスを与えるようになっている上記データベースへのファイルシステムインタフェースと、を備えていることを特徴とするシステム。

【請求項３２】　上記ファイルシステムをベースとするアプリケーションは、データベース・アンアウェアである請求項３１に記載のシステム。

【請求項３３】　上記データベースは、オブジェクト・関係型データベースからなる請求項３２に記載のシステム。

【請求項３４】　上記データベースは、関係型データベースからなる請求項３２に記載のシステム。

【請求項３５】　上記データベースは、オブジェクト指向データベースからなる請求項３２に記載のシステム。

【請求項３６】　上記データベース内の情報を管理するためのデータベース管理システムを更に備えている請求項３３に記載のシステム。

【請求項３７】　上記データベース上でデータベース動作を遂行するためのデータベースエンジンを更に備え、上記ファイルシステムインタフェースはファイルシステム要求を受信し、これらの受信した要求を上記データベースエンジンが理解できる形状に変形する請求項３１に記載のシステム。

【請求項３８】　上記ファイルシステムに導かれるファイルシステム要求を、上記ファイルシステムインタフェースに導かれるファイルシステムから区別するためのモジュールを更に備えている請求項３１に記載のシステム。

【請求項３９】　上記ファイルシステムインタフェースは、ファイルオブジェクトを含む拡張モジュールを更に備えている請求項３１に記載のシステム。

【請求項４０】　上記ファイルオブジェクトは、データベースオブジェクトをファイルシステムオブジェクトに変換するための情報からなる請求項３９に記載のシステム。

【請求項４１】　上記ファイルシステムインタフェースは、データベースオブジェクトの異なる集合をファイルシステムオブジェクトに変換するための情報を各々が含む複数の拡張モジュールからなる請求項３１に記載のシステム。

【請求項４２】　上記ファイルシステムインタフェースは、単一のデータベースオブジェクトを複数のファイルシステムオブジェクトに変換するための情報を各々が含む複数の拡張モジュールからなる請求項３１に記載のシステム。

【請求項４３】　コンピュータ可読媒体上に存在し、ファイルシステムをベースとするアプリケーション及びデータベースからなるデータベースリポジトリ管理システムのためのコンピュータソフトウェアであって、上記コンピュータプログラムはコンピュータシステムに、
上記ファイルシステムをベースとするアプリケーションによって発行されたファイルシステム要求を受信し、
上記ファイルシステム要求をデータベース動作に変換し、
上記データベース上で上記データベース動作を遂行することによって、上記データベースから情報を検索し、
上記検索された情報を、所定の基準に従って１つまたはそれ以上のファイルシステムオブジェクトに変形し、そして上記１つまたはそれ以上のファイルシステムオブジェクトを、上記ファイルシステムをベースとするアプリケーションに提示する諸動作を遂行せしめることを特徴とするコンピュータソフトウェア。

【請求項４４】　データベース内の情報にアクセスするためにコンピュータシステム上で遂行される方法であって、
データベースオブジェクトを、データベース・アンウェアアプリケーションがファイルシステムオブジェクトとして使用可能にするステップと、
上記使用可能なファイルシステムオブジェクトに対応するファイルシステム要求を、上記データベース・アンウェアアプリケーションから受信するステップと、
上記データベース・アンウェアアプリケーションからの上記ファイルシステム要求を、データベース動作に変形するステップと、
上記データベース動作を、データベースエンジンを使用して上記データベース上で遂行するステップと、
上記データベース動作に応答して、上記データベースから情報を検索するステップと、
上記データベースオブジェクトに関連する情報を、ファ

イルシステムオブジェクトとして上記データベース・アンウェアアプリケーションに戻すステップと、を備えていることを特徴とする方法。

【請求項４５】　データベース内の情報にアクセスするコンピュータをベースとする方法であって、

ファイルハンドルを、データベース内のデータベースオブジェクトを指定する情報を用いてエンコードするステップと、

アプリケーションによって発行されたファイルシステム要求に応答し、上記エンコードされたファイルハンドルを伝送するステップと、

上記受信したファイルハンドルをデコードし、上記ファイルシステム要求に関連する上記データベースオブジェクトを識別するステップと、を備えていることを特徴とする方法。

【請求項４６】　上記エンコーディングは、ＮＦＳプロトコルに基づいている請求項４５に記載の方法。

【請求項４７】　上記エンコーディングは、上記発行されたファイルシステム要求に対応する拡張モジュールを識別する情報を、上記ファイルハンドル内に含ませることからなる請求項４５に記載の方法。

【請求項４８】　上記エンコーディングは、上記発行されたファイルシステム要求に対応するデータベース表及び行を識別する情報を、上記ファイルハンドル内に含ませることからなる請求項４５に記載の方法。

【請求項４９】　上記エンコーディングは、上記発行されたファイルシステム要求に対応するメタデータを識別する情報を、上記ファイルハンドル内に含ませることからなる請求項４５に記載の方法。

【請求項５０】　上記エンコーディングは、上記発行されたファイルシステム要求に対応する上記データベースオブジェクトを指す情報を、上記ファイルハンドル内に含ませることからなる請求項４５に記載の方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、データベース内の情報へのアクセスに関する。

【０００２】

【従来の技術】データベースは、「データベース・エンジン」（即ち、データベース内のデータを検索、または操作するためのソフトウェア方法の集まり）によってコヒーレントな手法で検索、格納、及び探索することができるように、論理的に編成された情報の本体である。一般にデータベースは、関係型データベース、オブジェクト指向データベース、及びオブジェクト・関係型データベースの３つのカテゴリに分類される。関係型データベース（ＲＤＢ）は、データベース開発者が選択した何れかの手法で、互いに実質的に関係付ける（即ち、「結合」する）ことができる固定フィールド二次元表の集まりである。関係型データベースの構造は、表間の関係を選択的に再定義することによって変更することができる。データベース・エンジンは、構造化照会言語（ＳＱＬ）によって、または他のメカニズムによって表される方法の何れかのような種々のデータベース照会プロトコルの何れかを使用することによって、関係型データベース上の複雑な探索を迅速、且つ容易に遂行することができる。表間の関係によって、探索の結果を、データベース内の他の表内の対応情報と自動的に相互参照することができる。図１に示すように、関係型データベース１００は、例えば顧客表１０２を含み、顧客表１０２は論理リンク１０３によって注文表１０４に結合され、注文表１０４は論理リンク１０５によって在庫表１０５に結合されている。ユーザは、例えばしきい値よりも高い全ての注文番号に関してデータベース１００に照会することができる。注文表１０４は顧客表１０２及び在庫表１０６に結合されているから、照会によって得られた注文番号のリストを検索し、識別された注文番号に対応するそれぞれの顧客名及び在庫品目と共に表示することができる。

【０００３】オブジェクト指向データベース（ＯＯＤＢ）は、「オブジェクト」（即ち、データと、そのデータを操作するための規則とを収納しているソフトウェア要素）の集まりである。文字型のデータを格納することができる関係型データベースとは対照的に、ＯＯＤＢは実質的にどのような型のデータ（テキスト、３Ｄ図形イメージ、ビデオクリップス等）をも格納することができる。ＯＯＤＢは、その構成オブジェクトを関連規則と共にクラスのある階層内に格納するので、ＯＯＤＢは、それが有用な作業を行うのに必要な多くの論理を収納している。これに対して、関係型データベースはデータのみを収納しており、データを用いて有用な機能を遂行するためには外部アプリケーションソフトウェアに頼らなければならない。

【０００４】オブジェクト・関係型データベース（ＯＲＤＢ）は、他の２つの型のハイブリッドである。非文字データ（例えば、イメージファイル）を、大きい２進オブジェクト（ＢＬＯＢ：即ち、分化されていない大量のデータ）として、ＯＲＤＢ内に格納し、検索することができる。ＢＬＯＢ内に収納されているデータ（例えば、イメージファイルを見るためのユーティリティ）は、特定のＯＲＤＢインプリメンテーションに依存して、データベース内、またはその外部の何れかに格納することができる。Informix（登録商標）Universal Server（ＩＵＳ（登録商標））がオブジェクト・関係型データベース管理システム（ＯＲＤＢＭＳ）の例であり、これはＢＬＯＢを操作するための規則を内部に格納し、従ってそれらを「ネイティブ」データ型（即ち、ＯＲＤＢＭＳ自体が操作する能力を有しているデータ型）として取り扱うことができる。

【０００５】関係型、またはオブジェクト・関係型デー

タ内の情報は、典型的に、特定の機能を達成するために書かれたＳＱＬコンプライアントコンピュータプログラムによってアクセスされる。例えば、ユーザは、顧客の情報を格納しているデータベースから顧客のリストを検索するＳＱＬプログラムを書くことができる。代替として、データベース照会を支援する異なる多くのアプリケーションプログラムが使用可能であり、これらによってユーザは、任意の基準の集合（例えば、支払い期限の過ぎた勘定を有する全ての州外の顧客名）を指定することによって、データベース照会を対話式にフォーミュレートすることができる。この型のアプリケーションプログラムは、ユーザのデータベース照会をデータベースエンジンに提示し、該エンジンは要求された情報をデータベースから検索する。これらのアプリケーションプログラムは、それらがデータベースと対話し、データベースを操作する能力を有していることから、「データベース・アウェア」（ database aware ）と呼ばれる。

【０００６】これに対して殆どのアプリケーションプログラムは、それらがデータベース内に格納されている情報にアクセスすることができないことを意味する「データベース・アンアウェア」（ database unaware ）である。そうではなく、データベース・アンアウェアアプリケーションは、情報を離散したファイル内に格納し、検索するために、Sun Microsystems, Inc.によって開発された Network File System（ＮＦＳ）のようなファイルシステムに頼っている。データベース・アンアウェアアプリケーションは、分離した各文書を、アプリケーションのユーザによって識別された別個のディスクファイル内に格納する。例えば図２において、ファイルシステム２００は、取付けられている２つのディスクドライブ、即ちラベル「ａ：」に写像されているドライブ２０２、及びラベル「ｂ：」に写像されているドライブ２０４を有している。「ａ：」及び「ｂ：」の各ドライブは、１つまたはそれ以上のディレクトリ（「ａ：」ドライブ２０２上の「ｄｏｃｓ」、「ｂ：」ドライブ２０４上の「ｄｉｒ１」及び「ｄｉｒ２」）を含んでいる。これらのディレクトリ自体は、サブディレクトリ（「ｄｉｒ１」内の「ｓｕｂｄｉｒ１」、「ｄｉｒ２」内の「ｓｕｂｄｉｒ２」及び「ｓｕｂｄｉｒ３」）を有することができる等々であり、実質的に何れかのレベルの階層ネスティングが可能である。ファイル２０６－２１２は、ファイルシステム内の種々の何れかのディレクトリまたはサブディレクトリレベルに存在することができる。ラベル「ａ：」及び「ｂ：」は、ファイルシステムの「ネーム空間」を表している。即ち、「ａ：」または「ｂ：」から始まる全てのファイル名経路は、ファイルシステムのネーム空間内にある。図２に示すように、例えば州外の顧客名をリストする文書は、ファイル名経路
ａ：＼ｄｏｃｓ＼ｃｕｓｔ＿ｏｕｔｓｔａｔｅ．ｔｘｔ
によって定義された位置の、ファイルシステムのネーム空間内に格納される。これは、型「ｔｘｔ」の「州外顧客」と命名されたファイル２１１が、ラベル「ａ：」に写像されたディスクドライブ２０２上の「ｄｏｃｓ」と命名されたディレクトリ内に格納されることを意味している。支払い期限を過ぎた勘定を有する顧客名をリストする別の文書は、ファイル名経路
ｂ：＼ｄｏｃｓ＼ｃｕｓｔ＿ｏｖｅｒｄｕｅ．ｔｘｔ
に位置する別のディスクファイル内に格納される。これら２つのファイルは、データベース内の表が関係付けられていないことから、関係付けられていない、または結合されていない分離した別個のエンティティである。

【０００７】

【発明の概要】本発明の一面においては、データベース内の情報は、例えば、データベース・アンアウェアアプリケーションのようなアプリケーションが、１つまたはそれ以上のデータベースオブジェクト（例えば、表または行）を、１つまたはそれ以上のファイルシステムオブジェクト（例えば、ディレクトリ、ファイル、またはリンク）として使用できるようにすることによって、コンピュータシステムを用いてアクセスされる。データベースは、関係型、オブジェクト・関係型、またはオブジェクト指向であることができる。もし複数のファイルシステムオブジェクトが使用可能にされていれば、それらはまとめて階層ファイルシステムを表すことができる。ファイルシステムオブジェクトに対応するアプリケーションによって発行されたファイルシステム要求は、データベースエンジンを用いてデータベースに対して遂行される、例えばＳＱＬ照会のようなデータベース動作に変形される。

【０００８】データベース動作の結果として検索されたデータベースオブジェクトに関連付けられた情報は、１つまたはそれ以上のファイルシステムオブジェクトにフォーマットし、アプリケーションへ戻すことができる。検索された情報の特定のフォーマッティングは、拡張モジュール内に定義することができる。この拡張モジュールは、ファイルシステム要求を、データベース照会に変形するような特定の手法を定義する情報を更に含むことができる。データベース照会のフォーマッティング、情報の検索、及びそれをファイルシステムオブジェクトにフォーマットすることを含むデータベース動作は、アプリケーションに対して透過的に遂行される。ファイルシステムオブジェクトを受信すると、アプリケーションはそれらを、例えばファイルシステムオブジェクトの図形表現として、コンピュータの表示画面に表示することができる。使用可能にされたデータベースオブジェクトは、異なるアプリケーションが理解できるフォーマットで、複数のファイルシステムオブジェクトとして提示することができる。反対に、単一のファイルシステムオブジェクトを、複数のデータベースオブジェクトに対応させることができる。

【０００９】別の面においては、コンピュータをベースとするデータリポジトリ管理システムは、情報のデータベース、データベースを操作するためのファイルシステムをベースとするアプリケーションプログラム、データベース内の情報へのアクセスを有するファイルシステムをベースとするアプリケーションを提供するデータベースへのファイルシステムインタフェース（それ以外に、データベース・アンアウェアであることもできる）を含む。データリポジトリ管理システムは、ファイルシステムをベースとするアプリケーションに加えて、またはその代わりに、データベース内の情報を管理するデータベース管理システムを更に含むことができる。データリポジトリ管理システムは、ファイルシステムインタフェースへ導かれるファイルシステム要求から、ファイルシステムへ導かれるファイルシステム要求を区別するためのモジュールを含むことができる。ファイルシステムインタフェースは、１つまたはそれ以上のオブジェクトを収納する１つまたはそれ以上の拡張モジュールを含むことができる。各ファイルオブジェクトは、データベースオブジェクトをファイルシステムオブジェクトに変換するための情報を含んでいる。

【００１０】別の面においては、データベース内の情報は、データベース内のデータベースオブジェクトを指定する情報を用いてファイルハンドルをエンコードすることによって、コンピュータシステムを用いてアクセスされる。アプリケーションによって発行されたファイルシステム要求に応答して、エンコードされたファイルハンドルが伝送され、次いでデコードされてそのファイルシステム要求に関連するデータベースオブジェクトが識別される。エンコーディングは、ＮＦＳプロトコルに基づくことができる。エンコードされた情報は、発行されたファイルシステム要求に対応する情報を含むことができ、拡張モジュール、データベース表及び行、メタデータ、データオブジェクトを指すポインタ、またはそれらの組合わせを識別する。

【００１１】説明中のファイルシステムインタフェースの長所は、以下に記載の１つまたはそれ以上を含むことができる。データリポジトリとしてのファイルシステムに頼っている、またはそれ以外にデータベース・アンアウェアである（即ち、データベース内のデータにアクセスすることはできない）アプリケーションは、透過的な手法でデータ内の情報にアクセスすることを可能にされる。これらのデータベース・アンアウェアアプリケーションは、データベーズ・アウェアアプリケーションと共に、及び他のデータベース・アンアウェアアプリケーションと共に、継ぎ目なしにデータを共用することができる。ＩＸＦＳの下では、データベースはあるアプリケーションにとって、そのアプリケーションが使用可能な他のファイルシステムと形状または文字が異ならない別のローカル、または遠隔ファイルシステムとして見え得る。アプリケーションのプログラムコードを変化させる必要はないが、データベースまたはデータベースエンジンが必要になる。その結果、データベース・アンアウェアアプリケーションのユーザには、典型的にはデータベース・アウェアツールに関連する時間及び費用をかけずに、データベース機能性が提供されることになる。

【００１２】システムアドミニストレータは、ある企業の内部で統一されたデータベースリポジトリ戦略を作成するのに、異種のデータ格納技術（例えば、ファイルをベースシとするステムとデータベースシステム）を組合わせるために、ＩＸＦＳシステムを使用することができる。企業が関係型またはオブジェクト・関係型モデルのデータ記憶装置へ移るに際して、受け継がれたデータリポジトリ内に存在するデータをデータベースへ容易に転送することができるので、これらのデータリポジトリに対する企業のそれまでの投資は守られる。更に、ＩＸＦＳはデータベース・アンアウェアアプリケーションを使用してデータベース内に格納されているデータベースを管理することができるようにするので、データベース・アンアウェアアプリケーションへの企業の投資が増強される。

【００１３】データベース・アンアウェアアプリケーションをデータベース内の情報にアクセスさせる能力は、ファイルシステムパラダイムの簡易さと、データベース操作技術の精緻さ及び有効性とを組合わせる。このケイパビリティは、例えばハイパーテキスト・トランスファー・プロトコル（ＨＴＴＰ）を使用して、ユーザが大きいデータ記憶装置へのアクセスをシークするようになっているインターネット・ワールドワイド・ウェブアプリケーションに特に有用である。ＵＲＬ（ユニフォーム・リソース・ロケータ）エンコードされた要求に応答して、外部アプリケーションにデータベースからデータベースを検索させるようになっているコモン・ゲートウェイ・インタフェース（ＣＧＩ）スクリプトに対して、ＩＸＦＳはこのような要求を、データベースエンジンによって直接、迅速、且つ透過的に実行させることができる形状に変換する。

【００１４】データベース内の表の任意の集まりを、種々のファイルシステムオブジェクトとして表す能力は、ソフトウェア開発者に貴重な、且つ柔軟性に富むツールセットを提供する。ＩＸＦＳが拡張可能な性質を有しているので、ＩＸＦＳを実質的にどのような型のアプリケーションにも合わせることができるから、データベースは、アプリケーションの他のファイルシステムオブジェクトと一貫性のあるファイルシステムオブジェクトの集まりとして見えるようになる。他の長所及び特色は添付図面に基づく以下の説明から明白になるであろう。

【００１５】

【実施例】有効性が持続しているデータを格納するためにデータベースを使用すると、データリポジトリとして

ファイルシステムを使用した場合には得られない幾つかの利点が得られる。データベースの構造、及びデータベース内の表間の内部関係によって、情報に対する高速、且つ任意に複雑な照会をデータベースに遂行させることができる。これに対してファイルシステムは、ファイルシステムによって管理されるファイル内の特定データ項目を探索するための標準データ照会メカニズムを有していない。データベースによって提供される他の特色（普通のファイルシステムは類似のものを有していない）は、明確な管理ポリシー、監査ケイパビリティ、透過的なデータ複製、ロギング機能、及び一貫したバックアップ及び復元手順を含む。

【００１６】データベースシステムをデータリポジトリとして使用するためには、専用データベース・アウェアアプリケーションのような比較的複雑且つ高価なツールが必要であり、精緻さのレベルが高く、エンドユーザによる訓練を必要とする場合が多い。これに対して、ファイルシステムは一般的に使用が簡単であり、安価であり、そして普及している。実質的に全てのコンピュータオペレーティングシステムは、有効性が持続しているデータを格納するために、アプリケーションによって使用することができるネイティブファイルシステムを提供する。有効性が持続している全データのほぼ 85 － 90 %が、データベース・アンアウェアアプリケーションによってファイルシステム内に格納される主たる理由はこれである。Informix（登録商標）File System （ＩＸＦＳ）インタフェースと称する説明中のファイルシステムインタフェースは、データベース・アンアウェアアプリケーションが、アプリケーションに対して完全に透過であるようにデータベース内の情報にアクセス（即ち、読み出し及び書き込み）できるようにすることによって、コンピュータシステムのユーザに両世界に劣らないものを提供する。アプリケーションを、またはデータベースを変更する必要はない。ＩＸＦＳは、データベースの内容を、ディレクトリ、サブディレクトリ、ファイル、またはリンクのような「ファイルシステムオブジェクト」としてアプリケーションに提示する。これらのファイルシステムオブジェクトは、アプリケーションにとって、そのアプリケーションがデータを格納し、検索する通常のコースにおいて処理するファイルシステムオブジェクトと形状または文字が異ならないように見える。ＩＸＦＳは、そのデータベースの内容を表すファイルシステムオブジェクトに対して所望の動作を遂行することによって、データベース・アンアウェアアプリケーションのユーザがデータベースの内容にアクセスできるようにする。

【００１７】図3に示すように、ＩＸＦＳ　３００は、データベース・アンアウェアアプリケーション３０２とデータベース３０４との間に位置しており、アプリケーション３０２によってファイルシステム３０６に発行された全ての要求を監視する。アプリケーションがデータベース内の情報にアクセスするためにシークする場合、ＩＸＦＳシステムの構成要素は、ファイルシステム要求をデータベースが理解可能なデータベース照会フォーマットに変形する。同様に、データベースから受信した情報（例えば、アプリケーションによるファイルシステム読み出し要求に応答して）は、１つまたはそれ以上のファイルシステムオブジェクトとしてアプリケーションに提示される。以下に図４の流れ図を参照して、ＩＸＦＳの動作、及びそのコンピュータのオペレーティングシステムとの対話の高レベルの説明をする。ファイルシステム要求（例えば、データ読み出し、または書き込み）がアプリケーションプログラムによって発行されると、オペレーティングシステムは、それがＩＸＦＳによって管理されているファイルネーム空間内に収納されている情報と一致するか否かを決定する（ステップ４００）。オペレーティングシステムは、他のファイルシステム内に格納されているデータに対する要求と、ＩＸＦＳのファイルネーム空間内のデータに対する要求とを区別することができる。それは、データが、ファイルシステムのネーム空間（例えば、「ａ：」及び「ｂ：」）と相互に排他的なネーム空間（例えば、「ｘ：」）に写像されているからである。実際には、データベースは、オペレーティングシステムにとって、及びアプリケーションにとって、ラベル「ｘ：」に写像されたディスクドライブとして見える。

【００１８】もしファイルシステム要求がＩＸＦＳによって管理されている情報に向かって導かれなければ、要求は他のファイルシステムによって処理される（ステップ４０１）。一方、もしファイルシステム要求がＩＸＦＳのファイルネーム空間内の情報に一致していれば、オペレーティングシステムはその要求をＩＸＦＳに引渡し、ＩＸＦＳはその要求をＩＸＦＳの拡張可能な構成要素、即ち「拡張モジュール」へ供給してデータベースが理解できる形状（即ち、例えばＳＱＬ照会）に変形させる（ステップ４０２）。要求をデータベース照会に変形した後に、ＩＸＦＳ拡張モジュールはその照会をデータベースエンジンに提示し、該エンジンはそれを使用して所望の情報を変更することによって（書き込み要求の場合）、または所望の情報を検索してそれをＩＸＦＳへ戻す（読み出し要求の場合）ことによってデータベースにアクセスする（ステップ４０４）。もし情報がデータベースから検索されていれば（ステップ４０６）、ＩＸＦＳ拡張モジュールは所定の基準に従ってそれをファイルシステムオブジェクトにフォーマットし（ステップ４０８）、それらをアプリケーションに提示する（ステップ４１０）。ファイルシステムオブジェクトをＩＸＦＳから受信すると、アプリケーションはそれらを、それらがあたかもファイルシステムからきたように処理する。実際には、アプリケーションはアンアウェアであり、ファ

イルシステムオブジェクトはファイル以外の資源からきている。このように、ファイルシステムのネーム空間内のデータベースに対する全ての要求はファイルシステムによって処理され、一方データベースに割当てられているファイルネーム空間内のデータベースに対する全ての要求はＩＸＦＳによって処理される。

【００１９】データベースをアプリケーションへのファイルシステムとして表すために、ＩＸＦＳがどのように使用できるかの例を、図５－７に示す。ウィンドウをベースとするコンピュータシステムのユーザが、図２に示されているファイルシステム、及び図１に示されているデータベースの両方に格納されている情報を調べるために、フィールドシステムナビゲーションツールを使用するものとする。更に、ファイルシステムのネーム空間がラベル「ａ：」及び「ｂ：」によって表され、ＩＸＦＳがクライアントマシン上のドライブ「ｘ：」に写像されているものとする。図５に示すように、ナビゲーションツールウィンドウ５００は、最初にファイルシステムの２つのドライブ「ａ：」及び「ｂ：」、及びデータベースに対応するドライブ「ｘ：」を、詰まった状態で表示する。この点で、ユーザがドライブ「ｂ：」を拡張するようにナビゲーションツールに命令すると、図６に示すようにそのディレクトリ及びサブディレクトリの階層がユーザに対して表示されて２つのファイル、即ち「ｄｏｃ２０６．ｔｘｔ」及び「ｄｏｃ２０７．ｔｘｔ」を収納している「ｓｕｂｄｉｒ１」が開かれる。図６に表示されているファイル情報が、標準ファイルシステム動作を使用して「ａ：」及び「ｂ：」ドライブから検索される。

【００２０】次に、ユーザが、ＩＸＦＳを介してデータベースに写像されているドライブ「ｘ：」を拡張するようにナビゲーションツールに命令すると、ドライブ「ｘ：」の内容を調べることができる。ナビゲーションツールによって発行された対応するファイルシステム要求は、ドライブ「ｘ：」（データベースに割当てられたファイルネーム空間）を指しているから、ＩＸＦＳはファイルシステム要求を拡張モジュールに引渡すことによってそれを処理し、拡張モジュールは要求された情報をデータベースから検索するようにデータベース照会をフォーミュレートする。検索された情報は、ＩＸＦＳによって呼出された方法を用いてファイルシステムオブジェクトにフォーマットされ、ナビゲーションツールに戻される。データベースから検索された情報は、ナビゲーションツールにとって、及びナビゲーションツールのユーザにとっては、ファイルシステムで検索された他のファイルシステムオブジェクトと、文字に差はないように見える。図７に示すように、図１のデータベース１００内の表１０２、１０４、及び１０６が、３つの対応する位置、即ち「顧客」、「注文」、及び「在庫」に表示される。同様に、顧客表１０２内の３つの行（顧客＿名、顧客＿住所、及び顧客＿識別）は、「顧客」ディレクトリ内の３つの対応するサブディレクトリ（「名前」、「住所」、及び「識別」）内に表示される。「名前」サブディレクトリ内のエントリは、それぞれの内容（Adams、Andrews 、Brewster、等）について命名されたテキストファイルとして表示される。

【００２１】ユーザは、「ｘ：＼顧客＼名前」ディレクトリ内のどのテキストファイルも開くことができ（例えば、標準テキストエディタアプリケーションを用いて）、その内容を変更することができ、そして標準「ファイル保管」動作を遂行することができる。それに応答してＩＸＦＳは、それがデータベースに割当てられたファイルネーム空間に向けられたものであるので、ファイル保管要求を処理し、データベースの内容を適切に変更するために対応するデータベース動作をフォーミュレートする。ＩＸＦＳは、全てのファイルシステム動作をデータベース上で遂行することを許容する。例えば、ユーザは、ナビゲーションツールの適切な機能を使用して、「ｘ：＼ｃｕｓｔｏｍｅｒ」ディレクトリの名称を「「ｘ：＼ｃｕｓｔ」のような他の何等かに変更することができる。同様に、ユーザは、サブディレクトリまたは新しいファイルのような新しいファイルシステムオブジェクトを「ｘ：＼顧客」ディレクトリの下に作成することができる。更に、ファイルシステム内のファイルシステムオブジェクトを制限する（読み出し専用、隠し等）ことができるのと同じように、若干のユーザに対してデータベースの指定された部分へのアクセスを制限することもできる。

【００２２】ファイルシステム要求に応答してＩＸＦＳが戻すファイルシステムオブジェクトの特定の型、フォーマット、及び配列は、対応する拡張モジュール（データベースオブジェクト（例えば、表）の任意の集まりをカプセル封じするのに望ましいように合わせ、それらをファイルシステムオブジェクトの集まりとして表すことができるＩＸＦＳのソフトウェア構成要素）内に定義される。一実施例では、ＩＸＦＳは、ファイルシステム内のファイルをデータベースオブジェクトの集まりに１対１で写像する「基本拡張モジュール」（ＢＥＭ）を含んでいる。いろいろな使用に加えて、ＢＥＭは、ユーザが彼等のデータベースをファイルシステムからＩＵＳ（登録商標）データベース管理システム内へ迅速且つ透過的に移動させ、それに対してデータベース照会を走らせることを許容する。

【００２３】ＢＥＭは、ＩＸＦＳシステムを実現したソフトウェア開発者によって指定されたデータベース表の集まりをカプセル封じし、それらをファイルシステムオブジェクトとしてアプリケーションに提示することによって、ファイルシステムをエミュレートする。ＢＥＭによって指定された各表はディレクトリに対応し、表内の各行はディレクトリ内に存在するファイルシステムオブ

ジェクト（例えば、サブディレクトリ、ファイル、またはリンク）に対応する。ＢＥＭがカプセル封じする各データベース表毎に、ＢＥＭは図8に示すようなデータ構造を有する対応「ファイルオブジェクト」６００を含んでいる。ファイルオブジェクト６００は、ファイルシステム内のディレクトリ、ファイル、またはリンクに対応し、その直観的な表現を供給する。各ファイルオブジェクト６００は、ファイルオブジェクトの名前６０１（所与のディレクトリ内で独自のファイルシステムエンティティの識別子）と、型６０２（ディレクトリ、ファイル、またはリンク）と、所有権６０３（ファイルオブジェクトの所有者の識別子）と、アクセス権６０４（その所有者、コミュニティ、その他のオブジェクトへのアクセス権）と、時相特性６０５（最後の読み出し、書き込み、及びルックアップ動作のタイムスタンプ）と、人気度６０６（そのオブジェクトを指すリンクの数）と、サイズ６０７（バイトで表したオブジェクトのサイズ）とを含んでいる。ファイルオブジェクト６００は、その対応するデータオブジェクト６０８またはそのデータオブジェクトを指すポインタをも収納している。

【００２４】データベースの部分は、データベース表及び行を望むように選択することによって、及び選択された各表及び行が対応しているファイルシステムオブジェクトの型を指定することによって、ファイルシステム表現に写像される。例えば、図１のデータベースは、顧客、注文、及び在庫の各表が型「ディレクトリ」のＢＥＭ内の別個のファイルオブジェクトを占めるように指定することによって、図７に示すファイルシステム階層に写像されている。顧客表のファイルオブジェクト内では、名前、住所、及び識別の各行は型「ディレクトリ」として指定されており、それによって、階層的に優勢な顧客ディレクトリにとって、それらがサブディレクトリとして見えるようにされている。「顧客」表内の「名前」行内では、個々の顧客名エントリは、型「ファイル」としてファイルオブジェクト内に指定されており、図７に示すように、それらが個々のテキストファイルとして見えるようにされている。

【００２５】２つまたはそれ以上の異なるデータベースへの同時アクセスを提供するために、または同一のデータベース内の異なる情報にアクセスするために、または同一データベースオブジェクトの異なる解釈を提供するために、幾つかの異なるＩＸＦＳ拡張モジュールを存在させ、動作させることができる。単一の拡張モジュールは、同一の情報を複数の異なるフォーマット（例えば、異なる型のファイルシステムオブジェクト）で提示することができる。図７において、例えば、名前、住所、及び識別を含む顧客表は、「ｘｌｓ」フォーマットを理解する適切なスプレッドシートプログラムによって開くことができる単一のファイルシステムオブジェクト（例えば、全顧客の識別情報を含む「顧客．ｘｌｓ」と命名されているMicrosoft Excelファイル）として提示することができる。拡張モジュールは、「ｘ：＼顧客」ディレクトリ、その構成要素サブディレクトリ（「名前」、「住所」、「識別」）、及びその中に収納されているファイル（ Adams.txt、Andrews.txt 、Brewster.txt等）の代わりに、またはそれに加えて、「顧客．ｘｌｓ」ファイルオブジェクトをアプリケーションに提示するように構成することができる。

【００２６】別の例として、代替アプリケーションプログラムに使用させるために、拡張モジュールは「ｘ：＼顧客＼名前」内のテキストファイルを、幾つかの異なるフォーマットで提示するように構成することができる。図１のデータベースにおいては、例えば、単一のデータベースオブジェクトに関して複数のファイルシステムオブジェクトを提示することによって、複数の異なるファイルフォーマットを各顧客名毎に供給することができる。顧客 Adamsに関するデータベース表のエントリは、例えば、異なるフォーマット（ Microsoft Word と共に使用する場合の「Adams.doc」、Corel Wordperfect と共に使用する場合の「 Adams.wpd」、及びAdobe Framemaker と共に使用する場合の「 Adams.fm 」）を有する３つの別々のファイルシステムオブジェクトに写像することができる。「 Adams.doc」オブジェクトで情報を編集したユーザは、変化が自動的に「 Adams.wpd」、及び「 Adams.fm 」オブジェクトに反映されることを観測しよう。３つのファイルシステムオブジェクトの全てが同一のデータベースオブジェクト（即ち、顧客 Adamsのためのデータベースエントリ）に写像されるので、３つの代替ファイルシステムオブジェクトを互換的に使用して、Adams の情報の発散バージョンをもたらすような関係を持つことなく、顧客 Adamsに関する情報を見るか、または編集することができる。

【００２７】ソフトウェアライブラリから入手したのか、またはカストム使用に従って生成されたのかは拘わりなく、適切な拡張モジュールを使用することによってソフトウェア開発者は、データベース内に格納されている情報を検索するために、または、新しいまたは変更された情報をデータベース内に格納するために、データベース・アンアウェアアプリケーション（例えば、Microsoft Word、Microsoft Excel 、Lotus 1-2-3 ）を可能化することができる。同時に、データベース・アウェアアプリケーションは、データベース内に格納されている全ての情報（第１のインスタンスにおいてデータベース・アンアウェアによって格納された情報を含む）へのアクセスを続行することができる。これらのケイパビリティが一緒になって、単一企業規模のデータリポジトリが種々の異なるアプリケーション（データベース・アウェア及びデータベース・アンアウェアの両方）と共に維持されることを可能にし、それによってデータリポジトリ内の情報にアクセスできるようにする。更に、ＩＸＦＳ

は、異なるアプリケーション間（例えば、データベース・アウェアアプリケーションとデータベース・アンアウェアアプリケーションとの間、または2つの異種データベース・アンアウェアアプリケーションの間）のデータの移行を容易にする。

【0028】ＩＸＦＳシステムは、3つの異なるアーキテクチャ、即ちオブジェクトライブラリアーキテクチャ、核レベルマウンタブルファイルシステムアーキテクチャ、またはネットワークファイルシステムアーキテクチャによって実現することができる。第1のアプローチ、即ちオブジェクトライブラリアーキテクチャにおいては、データベース内の情報にアクセスする能力は、ライブラリ（例えば、MicrosoftWindows（登録商標）をベースとするプラットフォーム上のダイナミックリンクライブラリ（ＤＬＬ））を通してデータベース・アンアウェアアプリケーションに対して使用可能にされている1組のソフトウェアオブジェクトを通して達成される。データベースに対して動作するファイルシステムアクセス方法の一貫した集合を使用することによって、これらのソフトウェアオブジェクトは、ＡＮＳＩＣまたはＰＯＳＩＸ標準によって定義されている共通ファイルアクセス、アプリケーションプログラムインタフェース（ＡＰＩ）によって提供されるものに似た機能性を提供する。しかしながら、オブジェクトライブラリアーキテクチャを使用する場合には、ＩＸＦＳ関連オブジェクトの新しいライブラリと最初に再リンクされるＩＸＦＳシステムと共に使用される何等かのアプリケーションを必要とする。これに対して、他の2つのアーキテクチャは、既存のアプリケーションに何等の変更も施さずに、または再リンクすることなく、既存のアプリケーションがデータベース情報にアクセスすることを可能にする。

【0029】図9に示す核レベルマウンタブルファイルシステムアーキテクチャは、オペレーティングシステムレベルにおいてファイルシステム要求を横取りし、それらを処理のためにＩＸＦＳシステムに引渡す。核アーキテクチャにおいては、核アドレス空間７００は、使用中のオペレーテングシステム（例えば、UNIX, Windows（登録商標）NT）に特定のＩＸＦＳ核モジュール７０１を含むように変更される。アプリケーションプログラムからのファイルシステム要求は、ローカルクライアント７０５（ローカルホスト７０４のアドレス空間内に存在するアプリケーション）、またはＮＦＳ　７０３を介して遠隔顧客７０６（遠隔システムのアドレス空間内に存在するアプリケーション）の何れかから、ＯＳ核７０２によって受信される。

【0030】取付けられたファイルシステムデバイス（例えば、ディスクドライブ）のネーム空間に向けて導かれたファイルシステム要求は、ＮＦＳ　７０３によって普通の手法で処理される。データベースによって占められたネーム空間（ＩＸＦＳ核モジュール７０１によって決定される）に導かれたファイルシステム要求は、処理のためにＩＸＦＳデーモン７０８（または Windows NT の場合には、ＩＸＦＳ「サービス」）に引渡される。ＩＸＦＳデーモン７０８は、アクセスされる1つまたは複数のデータベース７０９への接続の管理、及びデータベースオブジェクトへのアクセスに使用されているファイル名のリストの維持を含む、幾つかの機能を遂行する。データベース７０９へのアクセス要求をＩＸＦＳ核モジュール７０１から受信すると、ＩＸＦＳデーモン７０８は要求を処理するために、適切なＩＸＦＳ拡張モジュール７０７を識別すべくファイル名ルックアップ手順を開始する。要求によって指定されたファイル名は、指定されたファイル名と各拡張モジュール内に収納されているファイルオブジェクトの名前のリストとを比較することによって、対応するＩＸＦＳ拡張モジュールのルックアップテーブル内への索引として使用される。どのＩＸＦＳ拡張モジュールが、指定されたファイル名と名前が一致するかを決定した後に、ＩＸＦＳデーモン７０８は要求をその拡張モジュール７０７へ転送する。拡張モジュールは、要求をデータベース７０９上で遂行されるデータベース動作に翻訳する。データベース照会動作に応答して生成されたどの情報も、ＩＸＦＳ拡張モジュール７０７によって呼出された方法によって、拡張モジュール内に定義されているファイルオブジェクト型に従ってファイルシステムオブジェクトにフォーマットされる。フォーマットされたファイルシステムオブジェクトは要求しているアプリケーションに提示される。

【0031】図10に示すネットワークアーキテクチャは、ネットワークレベルにおいてファイルシステム要求を横取りし、処理のためにそれらをＩＸＦＳシステムに引渡す。ネットワークアーキテクチャでは、ＯＳ核アーキテクチャ空間を変更する必要はない。そのようにはせず、ローカルクライアント８０５または遠隔クライアント８０６によって生成された全てのファイルシステムコマンドは、ループＮＦＳ接続８１０またはネットワークＮＦＳ接続８０３を介して、ＯＳ核の外側に存在するＮＦＳフロントエンドデーモン（またはサービス）８０４に引渡される。ＮＦＳフロントエンドデーモン８０４は、ＩＸＦＳデーモンの構成要素として実現されている。ファイルシステム要求を受信すると、ＮＦＳフロントエンドデーモン８０４はそれをＩＸＦＳデーモン８０８に、続いて適切なＩＸＦＳモジュール８０７に引渡す。ＩＸＦＳモジュール８０７は、上述した核レベルアーキテクチャのＩＸＦＳデーモン及びＩＸＦＳ拡張モジュールと同じ機能性を提供する。

【0032】ＩＸＦＳシステムは、関係型、オブジェクト・関係型、及びオブジェクト指向データベースを含むどのような型のデータベースのインタフェースをも提供するように、またＳＱＬに加えて、他のどのような型のデータベース照会プログラムも理解するように適合させ

ることができる。ＮＦＳは異なるプラットフォームにまたがってファイルを共用するために広く使用されている標準であるので、ＮＦＳが図１０のＩＸＦＳ実施例に使用されるネットワークプロトコルとして選択されたのである。しかしながら、ネットワークを通してファイルシステムへのアクセスを提供する他のどのようなネットワークプロトコル（例えば、Microsoft のコモンインターネットファイルシステム（ＣＩＦＳ）も、ＩＸＦＳネットワークアーキテクチャを実現するために使用することができる。

【００３３】ＮＦＳプロトコル（バージョン２）は、「ファイルハンドル」（ 32 バイト値）を使用して所望のファイルを識別することによって、クライアント及びサーバがファイル情報を交換することを可能にする。ＮＦＳがネットワークアーキテクチャにおけるネットワークファイルシステムプロトコルとして使用される場合には、ＮＦＳファイルハンドルを付加的に使用して、ＩＸＦＳがデータベースに対してファイルシステム要求を遂行するのを高速且つ効率的に達成させることができる。このようにするために、ＩＸＦＳは、ＮＦＳファイルハンドルを図１１に示すように、その動作に特定の情報を用いてエンコードする。ファイルハンドルのバイト１－８は、ＩＸＦＳ「マジックストリング」を保持している。このＩＸＦＳ「マジックストリング」は、ＩＸＦＳファイルハンドル（即ち、データベースに割当てられたファイルネーム空間に導かれるファイルシステム要求）とＮＦＳファイルハンドル（即ち、ファイルシステムに割当てられたファイルネーム空間に導かれるファイルシステム要求）とを、ＩＸＦＳが区別できるようにするエンティティである。マジックストリングは８バイトのストリングであり、ファイルハンドルをＩＸＦＳファイルハンドルとして識別するために、各バイトには 16 進の値ＦＦが割当てられる。バイト９及び 10 は、考慮中のファイルハンドルを管理することをジョブとする特定のＩＸＦＳ拡張モジュールを識別する。このようにすると、ＩＸＦＳファイルハンドルをエンコードすることにより、空（「影」）のディレクトリ樹木をファイルシステム内に維持する必要性が排除される。このようにしない場合には、ＩＸＦＳシステムによって管理される情報に対応する標準ＮＦＳファイルハンドルを生成する必要があろう。同様に、このエンコーディング計画によって、ＮＦＳファイルハンドルとＩＸＦＳファイルハンドルとの間に独特な写像エンティティ（例えば、ルックアップテーブル）を維持する必要がなくなる。

【００３４】図１１のＮＦＳファイルハンドルの残余のバイトは、ファイルハンドルのバイト９及び 10 によって識別された拡張モジュールに特定の情報を含む。バイト 11 － 14 及び 15 － 18 はそれぞれ、ファイルハンドルに対応するデータベース表及び行を識別する。バイト 19 － 22 は、考慮中のファイルハンドルのためのファイル属性、及びデータベース内のデータを指すポインタを記述するメタデータに対応するｉノード（情報ノード）表及び行を識別する。現在はバイト 19 －32は未使用であるが、新たに開発される拡張モジュールが使用できるようにしてある。図１１に表されているエンコーディングの結果として、要素をデータベース内に配置できる効率が向上し、新しいＩＸＦＳ拡張モジュールの設計の複雑さが減少する。

【００３５】核レベル及びネットワークアーキテクチャは、互いに異なる利点を提供する。核レベルアプローチは、ファイルシステム要求の発行と、ＩＸＦＳからのファイルシステムオブジェクトの戻りとの間のデータベース経路をより短くする点が２つよりも効率的である。一方、ネットワークベースアーキテクチャは、ネットワークアーキテクチャを実現するのにオペレーティングシステム核を変更する必要がないので、異なるプラットフォーム間でＩＸＦＳをポートするために必要な努力を大幅に減少させる。しかしながら、両方の場合共、ＩＸＦＳシステムは、データベース内にデータへの透過的なアクセスを有するデータベース・アンアウェアアプリケーションを提供しながら、有効性が持続されているデータ格納のためにデータベースを使用するという固有の長所を維持する。

【００３６】上述した方法及びメカニズムは、どのような特定のハードウェアまたはソフトウェア構成にも制限されるものではなく、むしろそれらはデータベース操作を遂行することができるどのような計算、または処理環境にも適用性を見出すことができる。上述した技術は、ハードウェアまたはソフトウェア、または両者の組合わせで実現することができる。これらの技術は、プロセッサ、プロセッサ可読記憶媒体（揮発性及び不揮発性メモリ及び／または記憶素子を含む）、及び適当な入力及び出力デバイスを各々が含むプログラマブルコンピュータ上で実行されるコンピュータプログラムで実現することが好ましい。プログラムコードが入力デバイスを使用して入力されるデータに適用され、上述した機能が遂行され、そして出力情報が生成される。出力情報は、１つまたはそれ以上の出力デバイスに供給される。

【００３７】各プログラムは、コンピュータシステムと通信するために、高レベル手続き、またはオブジェクト指向プログラミング言語で実現することが好ましい。しかしながら、もし望むならば、プログラムはアセンブラ言語または機械語で実現することもできる。何れの場合も、言語はコンパイル済みの、即ち翻訳済みの言語であることができる。このような各コンピュータプログラムは、汎用または専用プログラマブルコンピュータによって可読の記憶媒体またはデバイス（例えば、ＣＤ－ＲＯＭ、ハードディスク、または磁気ディスケット）上に格納し、本明細書に記述した手順を遂行するために記憶媒体またはデバイスがコンピュータによって読まれた時に

コンピュータを構成し、動作させることが好ましい。システムは、コンピュータプログラムを用いて構成されたコンピュータ可読記憶媒体として実現され、この記憶媒体が特定の、及び所定の手法でコンピュータを動作させるように構成されているものと考えることもできる。

【００３８】他の実施例も特許請求の範囲の範囲内である。

【図面の簡単な説明】

【図１】関係型データベースの図である。

【図２】ファイルシステムの図である。

【図３】ファイルシステム及びデータベースにおいてデータにアクセスするためのシステムの図である。

【図４】データベース内のデータへ、図３のシステムを使用するアクセスの流れ図である。

【図５】ファイルシステム及びデータベースにおいて情報にアクセスするアプリケーションからの画面表示の例である。

【図６】図５の次の画面表示の例である。

【図７】図６の次の画面表示の例である。

【図８】ファイルオブジェクトのデータ構造図である。

【図９】核レベルファイルシステムアーキテクチャの図である。

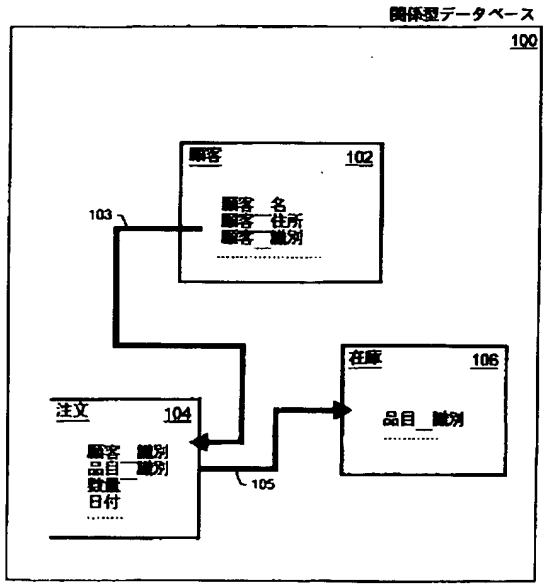
【図１０】ネットワークファイルシステムアーキテクチャの図である。

【図１１】図１０のネットワークファイルシステムアーキテクチャに使用されるＮＦＳファイルハンドルのデータ図である。

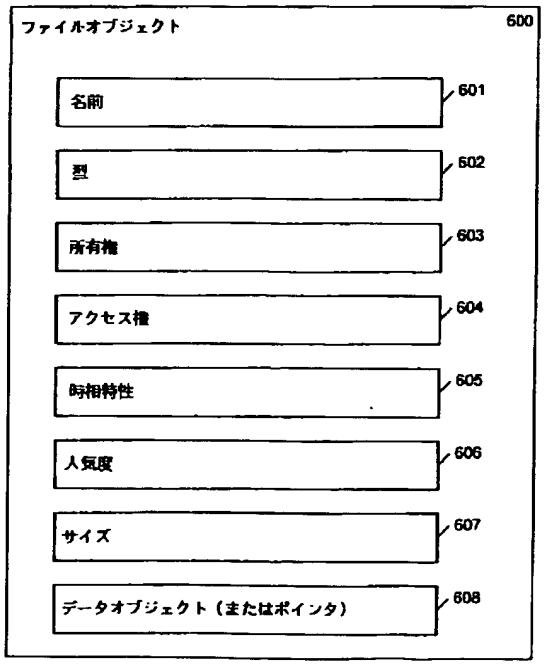
【符号の説明】

１００　関係型データベース
１０２　顧客表
１０３、１０５　論理リンク
１０４　注文表
１０６　在庫表
２００　ファイルシステム
２０２、２０４　ディスクドライブ
２０６－２１２　ファイル
３００　ＩＸＦＳ
３０２　データベース・アンウェアアプリケーション
３０４　データベース
５００　ナビゲーションツールウィンドウ
６００　ファイルオブジェクト
７００　核アドレス空間
７０１　ＩＸＦＳ核モジュール
７０２　オペレーティングシステム核
７０３　ＮＦＳ
７０４　ローカルホスト
７０５、８０５　ローカルクライアント
７０６、８０６　遠隔クライアント
７０７、８０７　ＩＸＦＳ拡張モジュール
７０８、８０８　ＩＸＦＳデーモン
７０９　データベース
８０３　ネットワークＮＦＳ接続
８０４　ＮＦＳフロントエンドデーモン
８１０　ループＮＦＳ接続

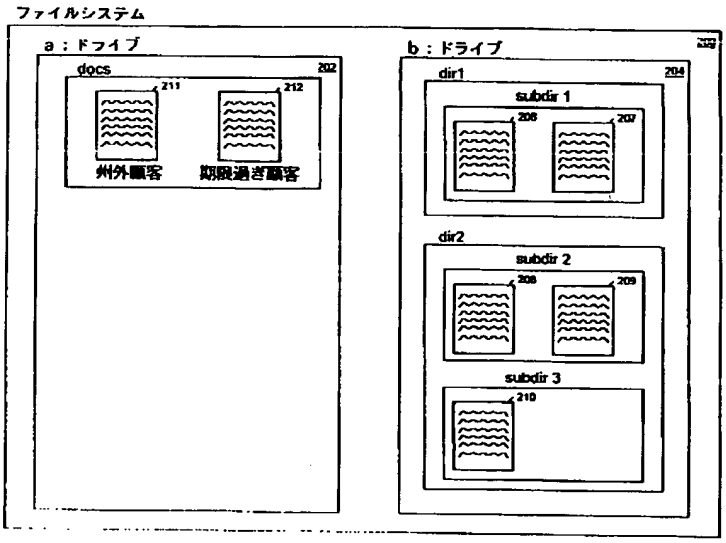
【図１】



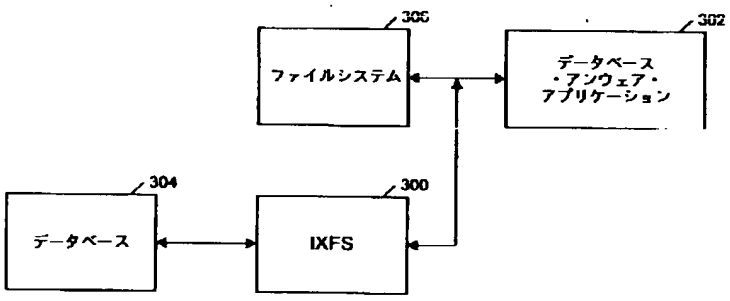
【図８】



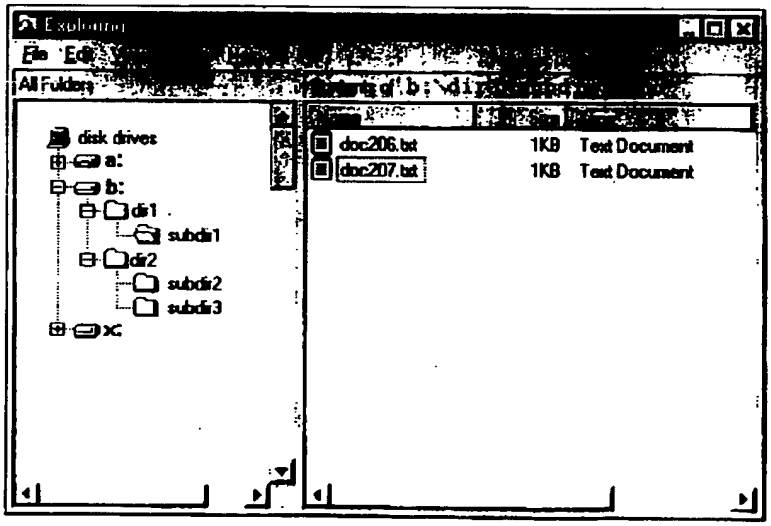
【図2】


ファイルシステム
a：ドライブ
docs
202
211
212
州外顧客
期限過ぎ顧客
b：ドライブ
dir1
204
subdir 1
206
207
dir2
subdir 2
208
209
subdir 3
210

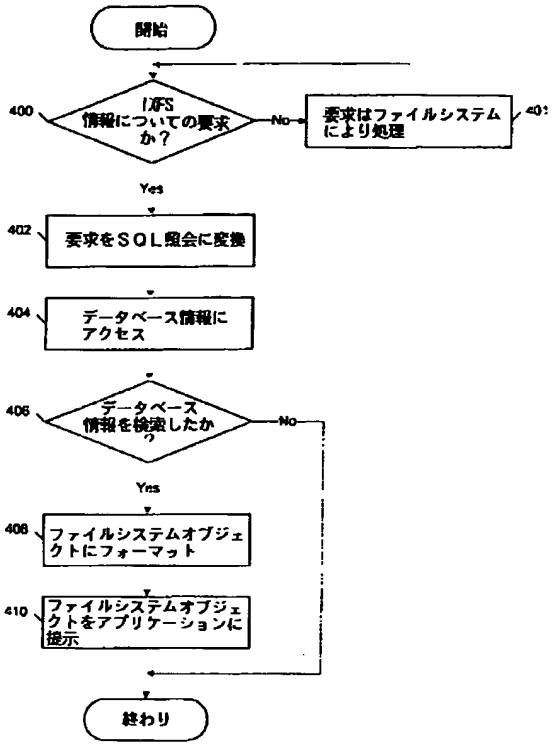
【図3】


305
ファイルシステム
302
データベース
・アンウェア・
アプリケーション
304
データベース
300
IXFS

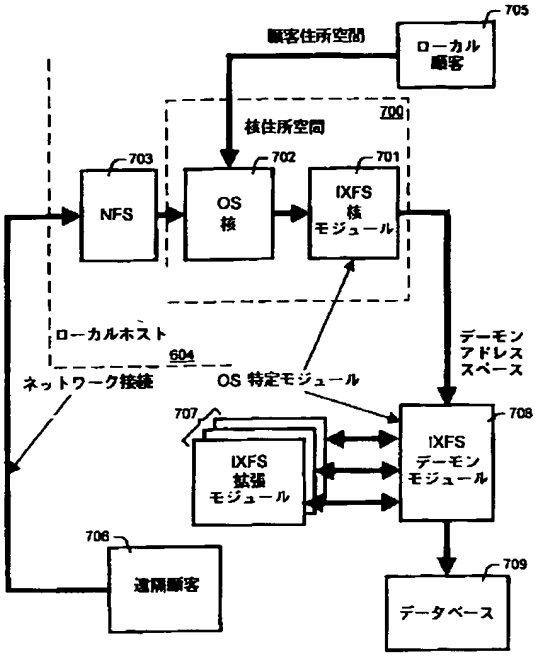
【図6】


File Edit
All Folders
disk drives
a:
b:
dir1
subdir1
dir2
subdir2
subdir3
c:
doc206.txt
1KB Text Document
doc207.txt
1KB Text Document

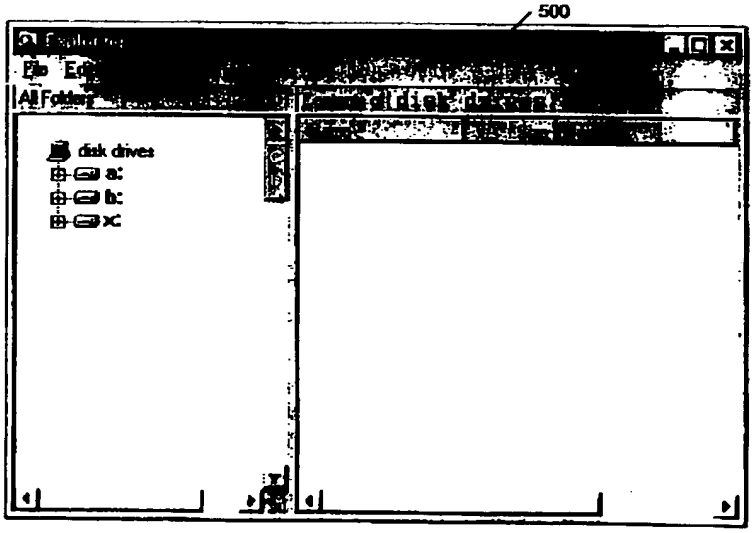
【図4】


開始
IXFS
情報についての要求
か？
400
No
要求はファイルシステム
により処理
401
Yes
402
要求をSQL照会に変換
404
データベース情報に
アクセス
406
データベース
情報を検索したか
？
No
Yes
408
ファイルシステムオブジェ
クトにフォーマット
410
ファイルシステムオブジェ
クトをアプリケーションに
提示
終わり

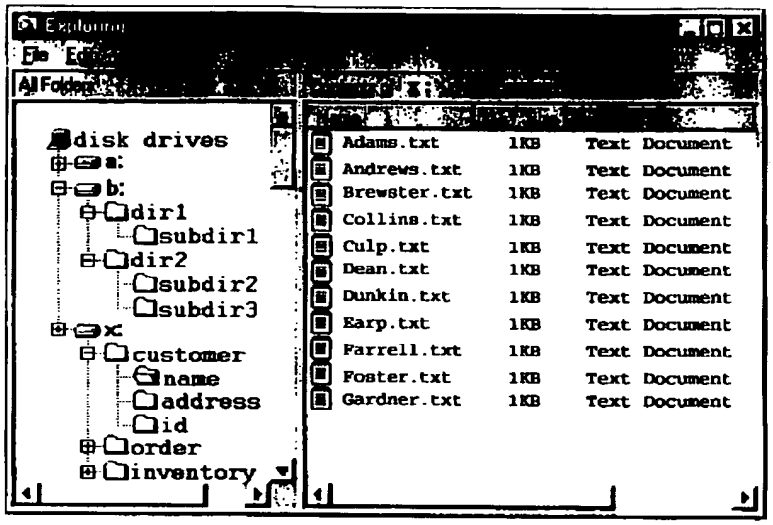
【図9】


705
顧客住所空間
ローカル
顧客
核住所空間
700
703
NFS
702
OS
核
701
IXFS
核
モジュール
ローカルホスト
604
デーモン
アドレス
スペース
ネットワーク接続
OS 特定モジュール
707
IXFS
拡張
モジュール
708
IXFS
デーモン
モジュール
706
遠隔顧客
709
データベース

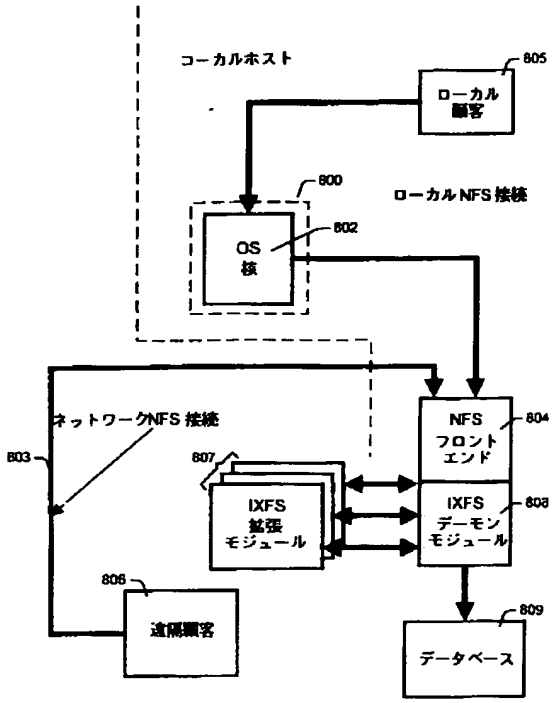
【図5】


500
File
All Folders
disk drives
a:
b:
c:

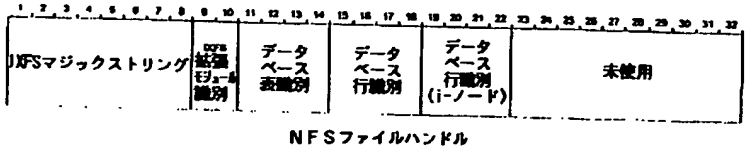
【図7】


All Folders
disk drives
a:
b:
dir1
subdir1
dir2
subdir2
subdir3
c:
customer
name
address
id
order
inventory
Adams.txt 1KB Text Document
Andrews.txt 1KB Text Document
Brewster.txt 1KB Text Document
Collins.txt 1KB Text Document
Culp.txt 1KB Text Document
Dean.txt 1KB Text Document
Dunkin.txt 1KB Text Document
Earp.txt 1KB Text Document
Farrell.txt 1KB Text Document
Foster.txt 1KB Text Document
Gardner.txt 1KB Text Document

【図10】


ローカルホスト
805
ローカル顧客
800
ローカルNFS接続
802
OS核
ネットワークNFS接続
803
807
NFSフロントエンド
804
IXFS拡張モジュール
IXFSデーモンモジュール
808
806
遠隔顧客
809
データベース

【図11】


IXFSマジックストリング
IXFS拡張モジュール識別
データベース表識別
データベース行識別
データベース行識別(i-ノード)
未使用
NFSファイルハンドル

フロントページの続き

(72)発明者　ラミアー　カンダサミー
アメリカ合衆国　カリフォルニア州
95014　クーパーティノ　ヴァーレイ　グリーン　20975－278

(72)発明者　ジョン　エイ　スキーアー
アメリカ合衆国　カリフォルニア州
95123　サン　ホセ　トノバー　ドライヴ　5698